IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CR Unsolicited Block ACK Part 2 | | | | |
| Date: 2018-06-28 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | email |
| Joe Andonieh | Peraso |  |  | joe@perasotech.com |
| Chris Hansen | Peraso |  |  | chris@covariantcorp.com |

Abstract

Proposed comment resolutions for CIDs 1102, 1220, 1768, 2242, 2270, 2132, 2134, 2284.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CID** | **Commenter** | **Clause** | **Comment** | **Proposed Change** |
| 1102 | Oren Kedem | 10.24.12.6 | For the first A-MPDU transmission for a specific tuple <TA, RA, TID>, the originator should transmit only a few MPDUs with the same tuple in the A-MPDU so as to synchronize WinStartR and WinStartB at the recipient    Please define better how much is few ? | 1 MPDU sent in A-MPDU aggregation |

Resolution: Revise

Either a single MPDU or a BAR can be used for synchronization. Clarifying text is added.

*Instruct the Editor to replace the first part of the first paragraph in D1.3 Section 10.25.10.6:*

~~For the first A-MPDU transmission for a specific tuple <TA, RA, TID>, the originator should transmit only a few MPDUs with the same tuple in the A-MPDU so as to synchronize WinStartR and WinStartB at the recipient.~~

with:

-- The originator shall not transmit an A-MPDU corresponding to a tuple <TA, RA, TID> before synchronizing the SSN, WinStartR, and WinStartB parameters for the specific tuple. The originator may transmit either a single MPDU or a BAR frame to initiate the synchronization. Synchronization is confirmed by reception of an Ack frame from the responder when a single MPDU is used, or by a Block Ack frame when a BAR is used.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1220 | Adrian Stephens | 10.24.12.5 | "control can be flushed" -- this is not the proper use of "can". Also the condition "when the STA stops receiving from the <TA, TID> pair" is not well defined. | Turn into a "may" statement, and define the condition for "stops receiving". |

Resolution: Reject

This comment has already been addressed in D1.2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1768 | Kazuyuki Sakoda | 10.24.12 | Unsolicited block ack extension looks to be mutually exclusive with SAR, EDMG flow control, or Multi-TID BA. However, there is no clear statement telling the fact. | Please describe brief introduction of the feature in clause 4, and clarify dependency of the additional BA protocols. |
| 2242 | Li-Hsiang Sun | 9.4.2.265 | When transmitting in an SP or in RD w/o AC contrain the STA could send PPDUs/A-PPDUs from multiple TIDs. Even if multi-TID AMPDU is not supported, there should still be a limitation on how many TIDs the receiver needs to maintain buffer in unsolicited BA extension | add number of TID limit field here and requirements in clause 10 |
| 2270 | Li-Hsiang Sun | 9.3.1.9.8 | The ack type '10' should not be used with unsolicited BA extension if the ack type is only used for TIDs w/o BA agreement | add a note indicating STA with unsolicited BA extension agreement shall not use this ack type or shall not use multi-TID BA |

Resolution: Revise

Discussion: Clause 4 does not have a description of the Block ACK mechanism and so clarification text should be added to section 10.25.10. The unsolicited Block ACK mechanism can optionally support Multi-TID, and a bit needs to be added to the Unsolicited Block ACK Extension element. Clarifying text is added as given below. Unsolicited Block ACK does not support segment and re-assembly and this is clarified. For EDMG flow control, the suggested resolution covers implementations that support the mandatory part of the EDMG flow control.

The reasoning for this decision is as follows. 1) The optional EDMG flow control feature requires signaling from both the originator and the recipient. 2) Using an ADDBA Request / Response, it is clear the direction from the originator to the recipient. However, this is not true when using the Association Request / Response mechanism. 3) A major change in the signaling architecture would be required to cover the proper negotiations and usages of the various optional features within the unsolicited Block ACK agreement. This would require a separate contribution.

*Instruct the Editor to modify Section 9.4.2.265 as shown:*

9.4.2.265 Unsolicited Block Ack Extension element

The Unsolicited Block Ack Extension element includes information necessary to set up an unsolicited block ack extension agreement between a non-AP and non-PCP STA and an AP or PCP at the association establishment, or between a pair of non-AP and non-PCP STAs using an Information Request and Information Response frame exchange. The format of the element is shown in Figure 74.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Element ID | Length | Element ID Extension | Unsolicited Block Ack Extension Parameters | Block Ack Timeout Value |
| Octets: | 1 | 1 | 1 | 2 | 2 |

Figure 80—Unsolicited Block Ack Extension element format

The Element ID, Length and Element ID Extension fields are defined in 9.4.2.1.

The Unsolicited Block Ack Extension Parameters field is defined in Figure 75.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | B0 B7 | B8 | B9 B15 | B16 B26 | B27 | B28 B31 |
|  | Reserved | A-MSDU Supported | Reserved | Buffer Size | Multi-TID  Supported | Reserved |
| Bits: | 8 | 1 | 7 | 11 | 1 | 4 |

Figure 81—Unsolicited Block Ack Extension Parameters field format

The A-MSDU Supported subfield is set to 1 to indicate that the STA supports an A-MSDU carried in a QoS Data frame sent under the unsolicited block ack extension. It is set to zero otherwise.

The Multi-TID Supported subfield is set to 1 to indicate that the STA supports multiple TIDs in unsolicited block ack agreements. An EDMG STA that advertises support for Multi-TID in an Unsolicited Block Ack Extension element also populates the EDMG Multi-TID Capability sub-field defined in 9.4.250.6.

An EDMG STA that does not advertise support for Multi-TID capability via the EDMG Multi-TID Capability sub-field defined in 9.4.250.6, sets the Multi-TID Subfield in an Unsolicited Block Ack Extension element to zero.

The Buffer Size subfield is an integer ranging from 1 to 1024 that indicates the number of buffers available for each unsolicited block ack extension agreement.

The Block Ack Timeout Value field is defined in 9.4.1.15.

*Instruct the Editor to add the following paragraph after the first paragrapgh in Section 10.25.10.6:*

The Multi-TID variant shall not be used by the originator if the recipient has not set the the Multi-TID Support bit in the Unsolicited Block Ack Extension Parameters field in the Unsolicited Block Ack Extension element.

*Instruct the Editor to add the following paragraph after the second paragrapgh in Section 10.63.1.*

An EDMG STA with dot11AMPDUwithMultipleTIDOptionImplemented set to true and that supports unsolicited block ack agreement may set the Multi-TID Supported subfield of the Unsolicited Block Ack Extension Parameters field of Unsolicited Block Ack Extension elements it sends to 1. Otherwise, the EDMG STA shall set the Multi-TID Supported subfield to zero to indicate that multi-TID is not supported under an unsolicited block ack agreement.

*Instruct the Editor to modify paragraph 3 in Section 10.62.1:*

**10.62.1 General**

Segmentation and reassembly is established for a particular TID only through the use of an ADDBA Request and ADDBA Response frame exchange that includes the SAR Configuration element. Segmentation and reassembly shall not be used under an unsolicited block ack agreement.

*Instruct the Editor to add this paragraph before the last paragraph in Section 10.25.2:*

**10.25.2 Setup and modification of the block ack parameters**

EDMG STAs that have established an unsolicited block ack agreement shall follow the RBUFCAP operation rules defined in 10.25.5.5 and 10.25.5.7.

*Instruct the Editor to add the following text to end of the last paragraph (before Table 35) in Section 10.25.5.5 that starts with “If an EDMG STA transmits a BlockAck frame in response to a BlockAckReq frame or an A-MPDU with Ack Policy equal to Normal Ack (i.e., implicit block ack request) during either full-state or …”:*

**10.25.5.5 Generation and transmission of BlockAck frames by an HT STA or DMG STA**

Within an Unsolicited Block ACK agreement, the RBUFCAP value used by an EDMG STA for the RBUFCAP calculation in Table 35 shall assume that the RBUFCAP Quantity Capability subfield value is "Not Supported" and shall not use a value between 1 and 254, inclusive.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2132 | Su Khiong Yong | 10.24.2 | "The originator does not receive an ADBBA Response frame with Status Code equal to SUCCESS prior to the reception of the BlockAck frame, or the tuple of the ADBBA Response frame with Status Code equal to SUCCESS received prior to the reception of the BlockAck frame is different than the BlockAck frame tuple. "    This paragraph first said orginator does not receive....and then using "or" to say that the orginator also does not receive the tuple of the ADBBA Response frame with Status Code equal to SUCCESS received prior to the reception of the BlockAck frame is different than the BlockAck frame. It is not clear what this sentence is trying to say | as suggested |

Resolution: Revise

Discussion: Commentor does not supply a solution, but we agree that the current text is confusing.

*Instruct the Editor to replace the paragraph in 10.25.2 Setup and modification of the block ack parameters as shown*:

* ~~The originator does not receive an ADBBA Response frame with Status Code equal to SUCCESS prior to the reception of the BlockAck frame, or the tuple of the ADBBA Response frame with Status Code equal to SUCCESS received prior to the reception of the BlockAck frame is different than the BlockAck frame tuple.~~
* Prior to the reception of the Block Ack frame, the originator does not receive an ADDBA Response frame with Status Code equal to SUCCESS with the same <TA, RA, TID> tuple as the Block Ack frame <TA, RA, TID> tuple.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2134 | Su Khiong Yong | 10.24.4 | What is "...at successful association establishment." means? | clarify |

Resolution: Revise

Discussion: Commentor does not supply a solution, but we agree that the current text is confusing.

*Instruct the Editor to replace the paragraph in 10.25.4 Receive buffer as shown*:

For each block ack agreement, the recipient maintains a MAC variable NextExpectedSequenceNumber. The NextExpectedSequenceNumber is initialized to the value of the Block Ack Starting Sequence Control field of the ADDBA Request frame of the accepted block ack agreement. Under an unsolicited block ack extension agreement, the NextExpectedSequenceNumber is initialized to zero upon ~~at~~ successful ~~association~~ establishment of the agreement. Under block ack agreement using segmentation and reassembly, the NextExpectedSequenceNumber is initialized to the value of the MSDU Starting Sequence Number subfield of the BAR Information field of the ADDBA Request frame that established the block ack agreement.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2284 | Li-Hsiang Sun | 10.24.12.3 | The procedure refers 10.24.7.3, but the step a) 1) in 10.24.7.3 requires SSN from ADDBA request | add a sentence "replace step a) 1) by setting WinStart\_R to 0" |

Resolution: Revise

Discussion: Agree that the text needs to be modified, but it should follow the resolution for CID 1102.

*Instruct the Editor to modify the text in 10.25.7.3 Soreboard context control during full-state operation as shown:*

1. At HT-immediate block ack agreement establishment:
2. In a block ack agreement that does not use segmentation and reassembly, *WinStartR* = *SSN* from the ADDBA Request frame that elicited the ADDBA Response frame that established the HT-immediate block ack agreement or from the MPDU that synchronized the unsolicted block ack agreement. Otherwise, *WinStartR = MPDU SSN* from the ADDBA Request frame that elicited the ADDBA Response frame that established the block ack agreement.
3. *WinEndR* = *WinStartR* + *WinSizeR* – 1